REMARKS

Applicant thanks Examiner for the detailed review of the application.

Claim Objections

The Office Action states:

3. Claim 14 is objected to because of the following informalities: "a switch handler to invoke a helper thread responsive to occurrence" should be changed to – a switch handler to invoke a helper thread responsive to the occurrence – in order to correct the grammar problem.

4. Claim 28 is objected to because of the following informalities: On line 9, "in response the user-defined trigger event" should be changed to – in response to the user-defined trigger event – in order to correct the grammar problem. Appropriate correction is required.

Applicant has amended claim 14 to "the occurrence," and claim 28 to "in response to the user-defined trigger event being detected," to correct the grammar informalities.

Claim Rejections - 35 USC § 112

The Office Action states:

5. Claims 18-27 and 29-30 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

Claim 18 has been amended to, "wherein the minimal context information includes a context weight less than a full context weight by at least a weight of excluding[[es]] traditional context information." As a result, the term minimal context information includes a weight less than a full weight by at least a traditional amount; embodiment of which are described in the applicant's specification at paragraphs 0041-0043.

Furthermore, claim 21 includes, "utilizing hardware to switch context information of the first thread with a second thread without operating system intervention, wherein the context

information has a first weight that is user-defined in a user-addressable control register." Here, the term minimal context has been removed, and applicant respectfully submits that context information having a first weight, which is user-defined in a user-addressable control register, particularly points out and distinctly claims the appropriate subject matter.

Additionally, the preambles of claims 29-30 have been amended from system to "processor" to provide the correct preamble reference.

Claim Rejections -35 USC § 102(b)

The Office Action states:

> **6. Claims 1, 4-7, 10-12, 14-16, and 18-20 are rejected under 35 U.S.C. 102(b) as being anticipated by Wang et al. (hereinafter Wang) (US 2002/0144083 A1).**

"[F]or anticipation under 35 U.S.C. 102, the reference must teach *every aspect* of the claimed invention ..." MPEP 706.02 (emphasis added). "The identical invention must be shown *in as complete detail as contained in the ... claim.*" *Richardson v., Suzuki Motor Co.*, 868 F. 2d 1226, 1236, 9 USPQ2d 1913, 1920 (Fed. Cir. 1989) (emphasis added).

Applicant's claim 1 includes, "a trigger-response mechanism that includes at least one bank of **user-programmable registers to identify a user-defined trigger event**." The Office Actions cites paragraph 0044 of Wang in relation to applicant's claim 1. However, paragraph 0044 of Wang simply discloses common architectural state registers (architecturally visible), i.e. general purpose registers, floating-point registers, predicate registers and control registers (See paragraph 0044). However, at no time does Wang disclose that these registers are user-programmable, but rather state they are "architecturally visible," i.e. privileged, not user-programmable. Furthermore, Wang does not disclose any of these registers or any other registers being utilized to "identify a user-defined trigger event."

Later, The Office Action also points to paragraph 0065 in reference to user-programamble registers to identify a user-defined trigger event. However, paragraph 0065 only discusses examining user-level control or data speculative calculations to determine success or failure. Paragraph 0066 provides an example, where a store conflicts with an earlier load, there is a branch into recovery code. Yet, at no time does Wang disclose user-programmable registers, or that those registers are utilized to define or specify a trigger event.

Similarly, applicant's claim 14 includes, "a user-addressable register to specify a user-defined trigger event based on the at least one raw event." As stated above, Wang does not disclose or include any registers that define trigger events. In contrast, Wang discloses marking instructions prior to a dynamic execution stream as a potential basic trigger (See paragraph 0055), but does not disclose any user addressable registers to specify user-defined trigger events, as in applicant's claim 14.

The Office action further states:

> **21. Claims 21-27 are rejected under 35 U.S.C. 102(e) as being anticipated by Kissell (US 2005/0050395 A1).**

Applicant's claim 21 includes, "utilizing hardware to switch context information of the first thread with a second thread without operating system intervention, wherein the context information has a first weight that is user-defined in a user-addressable control register." Note that Kissler merely discusses different types of threading in the back-ground section, such as when a processor supports two main threads. As a specific example, when a cache miss occurs on one main thread (212), then the resources of the processor, such as the execution units (multiply/divide unit 218), are potentially not wasting cycles, since another main thread (262) is still running and able to utilize unit 218 (see paragraph 0011).

However, note that there are a number of differences in applicant's claim 21 and Kissell.

First, the trigger condition is a "user-specified: trigger condition," while Kissell only discloses an example of the cache miss and does not describe the cache miss being a "user-defined" trigger condition. Second, Kissell's example in paragraph 0011, as described above, is in reference to the benefit of having two concurrently executing main threads, i.e. if one stalls the other continues to execute. In contrast, applicant's claim 21 includes performing a context switch in response to the user-specified condition from a first thread to a second thread, while Kissell's example does not include a context switch, but rather a continued operation of a second already active main thread (262). Third, Kissell does not disclose that a weight of the context switch being user-defined in a user-addressable control register.

Claim Rejections -35 USC § 103(a)

The Office Action states:

27. Claims 1-4 and 10-12 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ahmad et al. (hereinafter Ahmad) (US 7,010,672 B2) in view of Ranganathan (US 6,098,169).

"The examiner bears the initial burden of factually supporting any prima facie conclusion of obviousness." MPEP § 2142. It is well established that *prima facie* obviousness is only established when three basic criteria are met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. *In re Vaeck*, 947 F.2d 488 (Fed. Cir. 1991) (MPEP 2144). The Office Action has failed to meet one or more of these requirements.

First, applicant respectfully submits that there is no suggestion or motivation to combine

Ahmad and Ranganathan. Ahmad deals with providing breakpoints for program developers as part of an interactive debugging operation to scrutinize a developers current program, and does not suggest any performance monitoring or event detection for thread switch. In contrast, Ranganathan deals with monitoring performance during a thread switch, and only discusses monitoring of events as performance indicators during a thread switch for report back of performance metrics.

Second, applicant respectfully submits that the combination of Ranganathan and Ahmad do not disclose all of the elements of applicant's claims, such as claim 1. Applicant's claim 1includes, "**thread switch handler logic** coupled to the trigger-response mechanism **to perform a light-weight thread switch** from a first thread to a second thread **responsive to the user-defined trigger event occurring** during execution of the first thread," (amended claim 1 with emphasis added). As The Office Action states, Ahmad does not disclose any thread switch or the like. In addition, Ranganathan disclosure differs with applicant's claim 1 in a number of aspects.

First, Ranganathan's thread switch handler is implemented in software (see col.6 lines 5-10 → TSH 36 implemented through execution of device driver code; and col. 4 lines 37-46 → TSH 36 included in device driver 35 that is code executed at a Ring 0), not in hardware, i.e. thread switch handler logic, as in applicant's claim 1. Second, Ranganathan's thread switch handler is only to access values stored in performance registers after a thread switch has already occurred (see. Col. 6 liens 9-16 → TSH 36 receives indication that a thread switch **has taken place ...** accesses the values stored in the performance registers/counters). In contrast, applicant's thread switch handler logic in claim 1 is to **perform the thread switch** in response to **a user-defined trigger**, not obtain performance values after a switch has already occurred. In other words, there is no mention in Ranganathan that any trigger or event, such as the performance values, actually cause/trigger a thread switch, but rather those values are monitored and obtained once a thread switch occurs, which is in direct contrast to applicant's claim 1, which performs the thread switch in response to a

user-defined trigger.

The Office Action further states:

> **37. Claims 14, 16, and 18-20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Ahmad et al. (hereinafter Ahmad) (US 7,010,672 B2) in view of Marcuello et al. (hereinafter Marcuello) ("Thread-Spawning Schemes for Speculative Multithreading", IEEE, 2000).**

Applicant respectfully submits that there is no suggestion or motivation to combine Marcuello and Ahmad. As discussed above, Ahmad's disclosure deals with providing breakpoints for program developers as part of an interactive debugging operation to scrutinize a developer's current program, and does not suggest any wish or desire for improving processor parallelism or spawning new threads, which is the goal stated in the Office Action of Marcuello. In fact, a breakpoint for a developer is a complete system wide halt, so that a software developer may step through the code to determine actions of the machine, such as causes for errors. Therefore, the motivation to completely halt the system to inspect its state and a motivation to increase processing efficiency at a spawn point with a speculative thread is in direct contrast. Furthermore, it would be impermissible hindsight based on applicant's claims to combine Ahmad's disclosure of monitoring events to aide in breakpoint analysis for developers with Marcuello's motivation of spawning speculative threads to improve parallel processing.

The Office Action additionally states:

> **55. Claims 13, 17, and 28-30 are rejected under 35 U.S.C. 103(a) as being unpatentable over Wang et al. (hereinafter Wang) (US 2002/0144083 A1) in view of Spix et al. (hereinafter Spix) (US 6,195,676).**

Applicant's claim 28 includes, "user-programmable event logic coupled to the event detection logic to indicate a user-defined trigger event, the user-defined trigger event to be based on at least the raw event." As stated above in reference to independent claim 1, Wang does not disclose any logic, such as registers, that define a trigger event. For example, paragraph 0044 of Wang simply discloses common architectural state registers (architecturally visible), i.e. general purpose registers, floating-point registers, predicate registers and control registers (See paragraph 0044). However, at no time does Wang disclose that these registers are user-programmable, but rather state they are "architecturally visible," i.e. privileged, not user-programmable. Furthermore, Wang does not disclose any of these registers or any other registers or logic being utilized to "indicate a user-defined trigger event."

The Office Action also points to paragraph 0065 in reference to user-programamble logic to indicate a user-defined trigger event. However, paragraph 0065 only discusses examining user-level control or data speculative calculations to determine success or failure. Paragraph 0066 provides an example, where a store conflicts with an earlier load, there is a branch into recovery code. Yet, at no time does Wang disclose logic that indicates this or any other event is a trigger event.

As a result, applicant respectfully submits that independent claims 1, 14, 21, and 2, as well as their dependent claims, are now in condition for allowance for at least the reasons stated above.

If there are any additional charges, please charge Deposit Account No. 50-0221.

Furthermore, to expedite prosecution of the application, the Examiner is invited to contact David P. McAbee at (503) 712-4988 at any time to schedule a telephone interview in light of potential further rejection of the claims by the Examiner or questions regarding this amendment.

Respectfully submitted,
Intel Corporation

Dated: November 6, 2008

/David P. McAbee/Reg. No. 58,104
David P. McAbee
Reg. No. 58,104

Intel Corporation
M/S JF3-147
2111 NE 25th Avenue
Hillsboro, OR 97124
Tele – 503-712-4988
Fax – 503-264-1729